[illegible]

UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/666,425	09/19/2003	David E. Martin	2002P15891US01	8309

7590 04/26/2007

Siemens Corporation
Intellectual Property Department
170 Wood Avenue South
Iselin, NJ 08830

EXAMINER	
VERDI, KIMBLEANN C	
ART UNIT	PAPER NUMBER
2109	

SHORTENED STATUTORY PERIOD OF RESPONSE	MAIL DATE	DELIVERY MODE
3 MONTHS	04/26/2007	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

If NO period for reply is specified above, the maximum statutory period will apply and will expire 6 MONTHS from the mailing date of this communication.

Office Action Summary	**Application No.** 10/666,425	**Applicant(s)** MARTIN, DAVID E.	
	Examiner Kacy Verdi	**Art Unit** 2109	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.
- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on 19 September 2003.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) 1-38 is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) 1-38 is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☒ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on 19 September 2003 is/are: a)☒ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____ .
5) ☐ Notice of Informal Patent Application
6) ☐ Other: _____.

DETAILED ACTION

This office action is in response to the Application filed on September 19, 2003. Claims 1-38 are pending in the current application.

Claim Objections

1. Claim 38 is objected to because of the following informalities: claim 38, line 2, the recitation of "program code embodied in the medium" should be "program code stored on the medium". Appropriate correction is required.

Claim Rejections - 35 USC § 101

2. 35 U.S.C. 101 reads as follows:

> Whoever invents or discovers any new and useful process, machine, manufacture, or composition of matter, or any new and useful improvement thereof, may obtain a patent therefor, subject to the conditions and requirements of this title.

3. Claims 17-37 are rejected under 35 U.S.C. 101 because the claimed invention is directed to non-statutory subject matter.

With respect to claims 17-19, a "system for open development" is being recited; however, it appears that a system for open development would reasonably be interpreted by one of ordinary skill in the art as software, per se. A system for open development as claimed does not set forth a means to realize the software, per se such as being stored in a memory or computer storage media. As such, it is believed that a system for open development of claims 17-19 is reasonably interpreted as functional descriptive material, per se.

With respect to claims 20-32, a "system for providing an open development kit" is being recited; however, it appears that a system for providing an open development kit would reasonably be interpreted by one of ordinary skill in the art as software, per se.

A system for providing an open development kit as claimed does not set forth a means to realize the software, per se such as being stored in a memory or computer storage media to produce a tangible result. As such, it is believed that a system for providing an open development kit of claims 20-32 is reasonably interpreted as functional descriptive material, per se.

With respect to claims 33-37, a "kit for open development" is being recited; however, it appears that a kit for open development would reasonably be interpreted by one of ordinary skill in the art as software, per se. A kit for open development as claimed does not set forth a means to realize the software, per se such as being stored in a memory or computer storage media to produce a tangible result. As such, it is believed that a kit for open development of claims 33-37 is reasonably interpreted as functional descriptive material, per se.

Claim Rejections - 35 USC § 103

4. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

5. Claims 1-16 and 20-38 are rejected under 35 U.S.C. 103(a) as being unpatentable over United States Patent 6,282,454 to Papadopoulos et al. (hereinafter Papadopoulos) in view of United States Patent 5,974,470 to Hammond and further in view of United States Patent 5,634,114 to Shipley.

6. As to claims 1, 20, and 38 Papadopoulos teaches the invention substantially as claimed including a method and system of providing an open development kit, and computer program product, comprising the steps of:

sending a specific formatted load application request message (e.g. request sent from PLC 32, to back plane driver 56 of web server 30, Fig. 3) from a programmable logic controller (PLC) to an open development kit (ODK) subsystem (e.g. web server 30 or any other interface to translate PLC 32, Fig. 3, request message to different format, back plane driver 56, Fig. 3 receives request from PLC's ladder logic application 36, Fig.2, col. 4, lines 36-39); and

converting the specific formatted load application request message (e.g. client task request message) to a generic formatted load application request message (e.g. converted to HTTP message) (the server 20, Fig. 1, acts as the HTTP interpreter through TCP/IP stack 24, to interact with network interface 16 and application program 22 of PLC, Fig. 1, col. 3, lines 50-56);

sending the generic formatted load application request message from the ODK subsystem (e.g. web server) to an application (e.g. web browser) (the TCP/IP stack 24, enables data transfers between the application 22 and the user 2 through the internet 14, Fig. 1, utilizing IP protocol, col. 3, lines 53-60).

As to claim 38, Papadopoulos teaches the computer program product consisting of software program code (web server components (e.g. software modules) of Figure 3, are controlled by the real time operating system 44, which allocates CPU 46, Fig. 3 to various tasks (e.g. client 58, server 60, and HTTP 62 tasks, Fig. 3), message services and signals (e.g. CPU executes software instructions), col. 4, lines 49-53).

Although Papadopoulos teaches the invention substantially, Papadopoulos does not specifically disclose a request is for requesting an extension to be loaded and requesting execution of the application.

However Hammond teaches a request is for requesting an extension (e.g. DLL) to be loaded (system receives request for loading DLL in Windows™ operating system, col. 2, lines 36-39 of Hammond).

It would have been obvious to a person of ordinary skill in the art at the time the invention was made to have modified the PLC's Application Program request of Papadopoulos with the teachings of requesting an extension (e.g. DLL) to be loaded from Hammond because this feature would have provided any application access to a set of autonomous functions which could be linked to the application at run time per application request (col. 1, lines 30-34 and col. 5, 64-66 of Hammond).

In addition Shipley teaches requesting execution of the application (Compile/Link Application Program 36, and Load Application Program into Memory and execute it 38, Fig. 5).

As to claim 38, Shipley teaches a computer program product comprising a computer usable medium having readable program code (e.g. software modules of Web server) embodied in the medium (RAM 102 and ROM 103 for storing information and instructions (e.g. program code) for the processor 101, Fig. 6 (e.g. CPU), col. 9, lines 41-49).

It would have been obvious to a person of ordinary skill in the art at the time the invention was made to have further modified the PLC's Application Program request of Papadopoulos with the teachings of requesting an application to be executed from Shipley because these features would have provided an expected data interface between the DLL (e.g. requested extension) and the executable program (requested application) (col. 3, lines 64-65, of Shipley).

7. As to claims 2 and 21, Papadopoulos as modified teaches the method of claim 1 and the system of claim 20, further comprising the steps of:

initializing callback function pointers for use by the application as callback functions into the ODK subsystem (client application queues callback function associated with the request, col. 5, lines 49-51 of Papadopoulos); and

initializing (e.g. linking to application) the extension (e.g. DLL) after the extension (e.g. DLL) is loaded (DLL may be loaded and linked to an application at run time, col. 1, lines 31-32 of Hammond).

8. As to claim 3, Papadopoulos as modified teaches the method of claim 1, wherein the extension is a dynamic load library (DLL) (DLL consists of functions any application can use (e.g. extends the application functionality), col. 1, lines 33-34 of Hammond).

9. As to claims 4 and 22, Papadopoulos as modified teaches the method of claim 1 and the system of claim 20, further comprising the step of checking whether a stop to run transition has occurred in the PLC (user presses start box 168, Fig. 4, ladder logic diagram, on browser screen, col. 10, lines 8-12 of Papadopoulos) and if so, sending a specific formatted activate application message from the PLC to the ODK subsystem

(client task 58, allows an application to receive a new ladder logic MSTR request via the back plane driver 56, Fig. 3, col. 5, lines 36-38 of Papadopoulos).

10. As to claim 5 and 23, Papadopoulos as modified teaches the method of claim 4 and the system of claim 22, further comprising the step of calling an activate function in the application by the ODK subsystem (client task 58, allows an application to receive a new ladder logic MSTR request (e.g. activate function) via the back plane driver 56, Fig. 3, col. 5, lines 36-38 of Papadopoulos) thereby permitting scan cycle execution (e.g. test application executed, overload relay detects overload condition, col. 9, line 47 of Papadopoulos).

11. As to claims 6 and 24, Papadopoulos as modified teaches the method of claim 1 and the system of claim 20, wherein the requesting execution step includes checking whether there are requests for application execution in the PLC, (back plane driver detects new MSTR block request, col. 5, lines 51-52 of Papadopoulos) and if so, sending a specific formatted execution request (e.g. request sent from PLC to back plane of web server) from the PLC to the ODK subsystem (MSTR functions allow programs running in PLC 32, Fig. 3, to send commands (e.g. messages) to remote node on TCP/IP network (e.g. website), col. 5, lines 22-26, back plane driver 56, Fig. 3 receives request from PLC's ladder logic application 40, Fig.2, col. 4, lines 36-39 of Papadopoulos), converting the specific formatted execution request to a generic execution request message (e.g. converted to HTTP message) (the server 20, Fig. 1, acts as the HTTP interpreter through TCP/IP stack 24, to interact with network interface 16 and application program 22 of PLC, Fig. 1, col. 3, lines 50-56 of Papadopoulos), and

sending a generic execution request from the ODK subsystem to the application (the TCP/IP stack 24, enables data transfers between the application 22 and the user 2 through the internet 14, Fig. 1, utilizing IP protocol, col. 3, lines 53-60 of Papadopoulos).

12. As to claims 7 and 25, Papadopoulos as modified teaches the method of claim 6 and the system of claim 24, further comprising the step of executing the generic execution request by the application (Compile/Link Application Program 36, and Load Application Program into Memory and execute it 38, Fig. 5 of Shipley).

13. As to claims 8 and 26, Papadopoulos as modified teaches the method of claim 6 and the system of claim 25, further comprising the steps of:

sending a generic response from the application to the ODK subsystem (TCP/IP stack receives messages (e.g. response message from application) over Ethernet, col. 4, lines 55-57, and enables data transfer (e.g. communication) between the user 2 and the network interface 16, Fig. 1 over the internet, col. 3, lines 57-58 of Papadopoulos);

converting the generic response to a specific formatted response (TCP/IP stack 54 returns a response to the client task 58, Fig. 3, col. 6, lines 50-51 of Papadopoulos); and

sending the specific formatted response from the ODK subsystem to the PLC (client task 58, passes response to the back plane driver 56, Fig. 3, col. 6, lines 51-52, when response received back plane driver 56, Fig. 3passes it back to the PLC MSTR blocks, col. 5, lines 31-33 of Papadopoulos).

14. As to claims 9 and 27, of Papadopoulos as modified teaches the method of claim 8 and the system of claim 26, further comprising the step of returning at least one of

data and a control block from the application to the ODK subsystem (enables data transfer between the user 2, Fig. 1 and the website 4, Fig. 1, over the internet, col. 3, lines 57-58, the application calls a routine to pass the MSTR (blocks) response to the driver, col. 5, lines 57-59 of Papadopoulos), and from the ODK subsystem to the PLC (the driver passes back the response (MSTR blocks from above) to the ladder logic program of the PLC, col. 5, lines 29-30 and 59-61 of Papadopoulos).

15. As to claims 10 and 28, Papadopoulos as modified teaches the method of claim 1 and the system of claim 20, further comprising:

checking whether any requests for information are waiting in the application (Ethernet driver 46 places receive buffers (e.g. contain received messages from application) in the receive queue, on interrupt Ethernet driver 46 examines receive queue and if there are messages passes the receive queue to the TCP/IP stack 54, Fig. 3, col. 5, lines 4-6 of Papadopoulos), and, if so, requesting information from the PLC by the application (the server task 60, Fig. 3 allows an application to issue a request command to the PLC's executive program and receive a response (e.g. information), col. 5, lines 34-36 of Papadopoulos);

executing a function in the ODK subsystem specified by the application (based on the TCP/IP event (e.g. message received on TCP/IP stack from application), the server task 60, Fig. 3, uses the connection machine and transaction machine to advance the transaction (e.g. function), col. 7, lines 65-67 of Papadopoulos); and

performing a task in the PLC associated with the executed function in the ODK subsystem (server task 60 posts requests to the back plane driver 56, Fig. 3, col. 7, lines 42-43; the server task 60, Fig. 3 allows an application to issue a request command to the PLC's executive program and receive a response (e.g. information), col. 5, lines 34-36 of Papadopoulos).

16. As to claims 11 and 29, Papadopoulos as modified teaches the method of claim 10 and the system of claim 28, wherein in the requesting information step the application uses a call back pointer to generically request information (application queues both the request and call back function associated with the request, col. 5, lines 41-42 of Papadopoulos) and the executing step executes the function in the ODK subsystem corresponding to the callback pointer (when back plane driver 56, Fig. 3 services the request it calls the associated call back function, col. 5, lines 42-43 of Papadopoulos).

17. As to claims 12 and 30, Papadopoulos as modified teaches the method of claim 11 and the system of claim 29, in the executing a function step (call back function calls operating routine to pass message, col. 5, lines 45-46 of Papadopoulos), the function is provided by a dynamic link library (DLL for Windows™ operating system contains functions for user interface tasks of message sending, col. 1, lines 44-50 of Hammond).

18. As to claims 13 and 32, teaches the method of claim 10 and the system of claim 20, further comprising the step of returning a specific formatted response from the PLC to the ODK subsystem (server task 60 posts requests (to PLC) to back plane driver 56, and an associated call back routine sends the response to the server task 60, Fig. 3, col. 7, lines 42-45 of Papadopoulos), the ODK subsystem converting the specific formatted response to a generic response (when the response message is received the

server task 60, Fig. 3, finds connection and transaction machine in order to send response, col. 8 lines 3-5 of Papadopoulos), and returning the generic response from the ODK subsystem to the application (server task 60 uses TCP/IP stack 54 to transmit message, Fig. 3, col. 8, lines 6-7 of Papadopoulos).

19. As to claim 14, teaches the method of claim 10, wherein when the checking determines that there are no requests for information waiting, and further comprising:

waiting until the PLC transitions from a run state to a stop state (user presses Stop box 170, Fig. 4, ladder logic diagram, on browser screen, col. 10, lines 8-12 of Papadopoulos);

sending a deactivate request from the PLC to the ODK subsystem (client task 58, allows an application to receive a new ladder logic MSTR request via the back plane driver 56, Fig. 3, col. 5, lines 36-38 of Papadopoulos); and

calling a deactivate function in the application (client task 58, allows an application to receive a new ladder logic MSTR request (e.g. deactivate function) via the back plane driver 56, Fig. 3, col. 5, lines 36-38 of Papadopoulos).

20. As to claim 15, Papadopoulos as modified teaches the method of claim 14, wherein when a memory clear or PLC shutdown occurs (e.g. PLC stops) (user presses Stop box 170, Fig. 4, ladder logic diagram, on browser screen, col. 10, lines 8-12 of Papadopoulos), the step of calling a release function in the application (client task 58, allows an application to receive a new ladder logic MSTR request (e.g. release function) via the back plane driver 56, Fig. 3, col. 5, lines 36-38 of Papadopoulos) and unloading the extension (DLL) occurs (free DLL module associated with application, col. 8, lines

26-27, Free Module function removes mapping of DLL to application, steps 78 and 82, Fig. 4 of Hammond) .

21. As to claims 16, 31, and 37, Papadopoulos as modified teaches the method of claim 1, the system of claim 20, and the kit of claim 34, wherein in the sending a load application request from a PLC is one of a soft PLC (e.g. ladder logic application of PLC) (back plane driver 56, Fig. 3 receives request from PLC's ladder logic application 40, Fig.2, col. 4, lines 36-39, PLC application program includes ladder logic program for controlling I/O devices 40, Fig. 2, col. 4, lines 37-39 of Papadopoulos).

22. As to claim 33, Papadopoulos as modified teaches a kit for open development, comprising:

a means for receiving a specific formatted message from a programmable logic controller (PLC) (back plane driver 56, Fig. 3 receives request from PLC's ladder logic MSTR blocks, col. 5, lines 29-30 of Papadopoulos);

a means for converting the specific formatted message to a generic formatted message (e.g. converted to HTTP message) (the server 20, Fig. 1, acts as the HTTP interpreter through TCP/IP stack 24, to interact with network interface 16 and application program 22 of PLC, Fig. 1, col. 3, lines 50-56 of Papadopoulos); and

a means for sending the generic formatted message to an application for execution (the TCP/IP stack 24, enables data transfers between the application 22 and the user 2 through the internet 14, Fig. 1, utilizing IP protocol, col. 3, lines 53-60 of Papadopoulos).

23. As to claim 34 Papadopoulos as modified, teaches the kit of claim 33, further comprising:

a means for receiving a generic formatted message from the application (TCP/IP stack receives messages (e.g. response message from application) over Ethernet, col. 4, lines 55-57, and enables data transfer (e.g. communication) between the user 2 and the network interface 16, Fig. 1 over the internet, col. 3, lines 57-58 of Papadopoulos);

a means for converting the generic formatted message to a specific formatted message (TCP/IP stack 54 returns a response to the client task 58, Fig. 3, col. 6, lines 50-51 of Papadopoulos); and

a means for sending the specific formatted message to the PLC (client task 58, passes response to the back plane driver 56, Fig. 3, col. 6, lines 51-52, when response received back plane driver 56, Fig. 3, passes it back to the PLC MSTR blocks, col. 5, lines 31-33 of Papadopoulos).

24. As to claim 35, Papadopoulos as modified teaches the kit of claim 34, wherein at least one of the means includes using a dynamic link library that loads replaceable functionality (upgrade DLL to new code 24, compile/link to create new DLL 28, Fig. 4, of Shipley and col. 3, lines 55-56, of Hammond).

25. As to claim 36, Papadopoulos as modified teaches the kit of claim 34, wherein the generic formatted message is one of a response message (TCP/IP stack receives messages (e.g. response message from application) over Ethernet, col. 4, lines 55-57, and enables data transfer (e.g. communication) between the user 2 and the network interface 16, Fig. 1 over the internet, col. 3, lines 57-58 of Papadopoulos) and a request

for information message the server task 60, Fig. 3 allows an application to issue a request command to the PLC's executive program and receive a response (e.g. information), col. 5, lines 34-36 of Papadopoulos).

26. Claim 17 is rejected under 35 U.S.C. 103(a) as being unpatentable over United States Patent 6,282,454 to Papadopoulos et al. (hereinafter Papadopoulos) in view of United States Patent 6,121,924 to Meek et al. (hereinafter Meek).

27. As to claim 17, Papadopoulos teaches the invention substantially as claimed including teaches a system for open development, comprising:

one or more extensions (e.g. client task 58, Fig. 3) adapted for use in a real-time operating environment (real time operating system 44, Fig. 3); and

a CPU adapted to execute a programmable logic controller (PLC) application program in the real-time operating environment and adapted to execute the one or more extensions, wherein the one or more extensions (e.g. client task 58, Fig. 3) provide access into the scan cycle of the PLC (client task 58, allows an application to receive a new ladder logic MSTR request (e.g. activate function) via the back plane driver 56, Fig. 3, col. 5, lines 36-38, for example: test application executed, overload relay detects overload condition, col. 9, line 47 of Papadopoulos).

Although Papadopoulos teaches the invention substantially, Papadopoulos does not specifically disclose a virtual CPU adapted to provide replaceable functionality to the operation of the PLC.

However Meek discloses a virtual CPU (virtual CPU 402, Fig. 4 (also referred to as metadata engine), col. 8, lines 18-19) adapted to provide replaceable functionality (e.g. replacement routines) to the operation of the PLC (replacement routines 400 on run on a virtual CPU 402, Fig. 4).

It would have been obvious to a person of ordinary skill in the art at the time the invention was made to have modified the CPU of Papadopoulos with the teachings of a virtual CPU from Meek because this feature would have provided a mechanism to execute of replacement routines (e.g. extensions), which could have been used in various different hardware platforms because they are provided in platform-independent interpretive code, on a virtual CPU which uses the same instruction set in any system platform (col. 8, lines 13-24 of Meek) .

28. Claim 18 is rejected under 35 U.S.C. 103(a) as being unpatentable over United States Patent 6,282,454 to Papadopoulos et al. (hereinafter Papadopoulos) in view of United States Patent 6,121,924 to Meek et al. (hereinafter Meek) as applied to claim 17 above, and further in view of United States Patent 6,539,422 B1 to Hunt et al. (hereinafter Hunt).

29. As to claim 18, Papadopoulos as modified by Meek teaches the invention substantially as claimed including teaches the system of claim 17, further comprising:

a system block loader (back plane driver 56, Fig. 3) adapted to load system blocks (back plane driver 56, Fig. 3 receives request from PLC's ladder logic MSTR blocks, col. 5, lines 29-30 of Papadopoulos), the system blocks including at least one of a system function block, a system function, a system data block (MSTR blocks are functions which include reading and writing data and allow programs running on the PLC to send commands and receive responses, col. 5, lines 22-28 of Papadopoulos).

Although Papadopoulos as modified by Meek teaches the invention substantially, Papadopoulos as modified by Meek does not specifically disclose an ODK SB Add-on dynamic link library (DLL) for implementing a common object module (COM) interface for the virtual CPU and system block loader.

However Hunt discloses an ODK SB Add-on dynamic link library (DLL) for implementing a common object module (COM) interface for the virtual CPU and system block loader.

It would have been obvious to a person of ordinary skill in the art at the time the invention was made to have modified the back plane driver of Papadopoulos as modified by Meek with the teachings of a DLL for implementing a COM interface from Hunt because these feature would have provided a separate file a programmer may make connections to a module (e.g. client, server, HTTP task) without effecting the operation of the calling program or any other routine (col. 10, lines 5-8 of Hunt).

30. Claim 19 is rejected under 35 U.S.C. 103(a) as being unpatentable over United States Patent 6,282,454 to Papadopoulos et al. (hereinafter Papadopoulos) in view of United States Patent 6,121,924 to Meek et al. (hereinafter Meek) as applied to claim 17 above, and further in view of 5,974,470 to Hammond.

31. As to claim 19, Papadopoulos as modified by Meek does not teach the system of claim 17, wherein the one or more extensions are dynamic link libraries.

However Hammond teaches the system of claim 17, wherein the one or more extensions are dynamic link libraries (DLL consists of functions any application can use (e.g. extends the application functionality), col. 1, lines 33-34 of Hammond).

It would have been obvious to a person of ordinary skill in the art at the time the invention was made to have modified the web server client, server, and HTTP tasks of Papadopoulos as modified by Meek with the teachings of a dynamic link library from Hammond because this feature would have provided patches to standard Windows™ API call logic with code that serves to more accurately load the correct DLL's associated with specified applications (col. 3, lines 51-56, of Hammond).

Conclusion

32. The prior art made of record and not relied upon is considered pertinent to applicant's disclosure.

United States Patent 5,283,900 to Frankel et al. discloses a data processor, such as a digital signal processor, that has augmented memory, I/O and math units for real-time performance of complex functions, is placed under the control of a group of abstract object-oriented modules arranged with an underlying operational nucleus that includes a real-time kernel.

United States Patent 5,333,298 to Bland et al. discloses a computer system, having external peripherals, includes an operating system and application packages residing therein. Data exchange logic permits an outside application package (generated by a third party to perform a predefined application function in a general purpose computer environment) to be integrated into the computer system.

United States Patent 5,548,759 to Lipe discloses multiple files are combined into a single file (64a) in new executable format to operate a hardware or software device, such as a peripheral device (30a), while retaining compatibility with an operating system (40).

United States Patent 5,838,911 to Rosenhauer et al. discloses a method and system are disclosed for obtaining network information in a computer network.

United States Patent 6,334,158 B1 to Jennyc et al. discloses an application integrator that provides a user-interactive environment for integrating software applications having incompatible programming interfaces, such as production control and business enterprise systems.

United States Patent 6,044,407 to Jones et al. discloses a computer implemented interface for interfacing an equipment controller to an equipment manager each arranged to respond to a reproduced text messages according to a first and second protocol.

United States Patent 6,547,150 B1 to Deo et al. discloses an application development system for smart cards includes a computer system having a client development interface (CDI), a smart card having a smart card development interface (SCDI) and application program interface (API) functionality within the smart card operating system.

United States Patent 6,675,226 B1 to Nair et al. discloses a multi-port, multi-network interface allows desktop-type computers to be used in industrial control environments employing time critical network communications over multiple networks.

United States Patent 6,721,607 B2 to Brault discloses a programmable logic controller system including a private operating system that does not support a TCP/IP protocol, and a communication module connected to a private communication bus and a TCP/IP network, and at least one intelligent module.

United States Patent Application Publication 2002/0133635 A1 to Schechter discloses a method and system for interacting with devices having different capabilities.

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Kacy Verdi whose telephone number is (571) 270-1654. The examiner can normally be reached on Monday-Friday 7:30am-5:00pm EST..

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Xiao Wu can be reached on (571) 272-7761. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

KV
April 17, 2007

[illegible]
XIAO WU
SUPERVISORY PATENT EXAMINER